INSTITUTO FEDERAL Bahia Campus Vitória da Conquista	**LISTA DE EXERCÍCIOS 04**	
	CURSO: Bacharelado em Sistemas de Informação	**MODALIDADE:** Ensino Superior
	MÓDULO/SEMESTRE/SÉRIE: 3º	**PERÍODO LETIVO:** 2024.1
	DISCIPLINA: Linguagem de Programação II	**CLASSE:** 20241.3.119.1N
	DOCENTE: Alexandro dos Santos Silva	

INSTRUÇÕES

- Para resolução das questões abaixo, será admitido o uso apenas da sintaxe adotada para escrita de programas em Java.

1. Implemente uma classe utilitária que disponha de método estático **`main`**, no qual serão deverá ser executada sequência de operações que se segue abaixo:
   a) Instanciação de dois objetos da classe **`java.util.ArrayList<E>`**, para fins de manipulação de duas listas de nomes de animais;
   b) Entrada de 10 (dez) nomes de animais, a serem armazenadas na primeira lista instanciada;
   c) Localização e remoção, na primeira lista, de todos os nomes de animais com 5 (cinco) ou mais caracteres, inserindo-os na segunda lista;
   d) Listagem de nomes de animais armazenados em cada lista.

   Observação: para a identificação da quantidade de caracteres dos nomes, invoque o método **`length`** a partir de cada objeto *string*.

2. Readapte a implementação da classe utilitária da questão anterior, pela substituição de nomes de animais por números inteiros. Em relação ao critério de remoção após inserção dos números inteiros na primeira lista, considere aqueles em que o primeiro e o último dígito sejam idênticos (critério se aplica, por exemplo, ao número 7297, mas não ao número 1234).

   Observação: para a identificação de números em que o primeiro dígito seja idêntico ao último dígito, sugere-se aqui comparação entre o último dígito de cada número e seu respectivo reverso (número com dígitos em ordem inversa à ordem dos dígitos no número original); para o obtenção do reverso de um número inteiro, segue-se abaixo implementação de método com tal propósito:

```
01  public static int getNumeroReverso(int n) {
02      int ultimoDigito;
03      int nReverso = 0;
04
05      while (n != 0) {
06          ultimoDigito = n % 10;
07          nReverso = nReverso * 10 + ultimoDigito;
08          n = n / 10;
09      }
10
11      return nReverso;
12  }
```

3. Readapte a implementação da classe utilitária da questão 01 de modo que seja impedida inserção, na primeira lista, de um mesmo nome de animal em mais de uma ocasião (independentemente de seus caracteres estarem em minúsculo ou em em maiúsculo). Em caso de tentativa nesse sentido, quando da operação de cada entrada, deverá ser exigida nova entrada até que o nome a ser inserido seja distinto de todos os nomes já inseridos anteriormente.

4. Uma boa prática de programação, conhecida como **baixo acoplamento**, consiste no uso de interfaces, restringindo invocação de classes concretas que as implemente apenas durante a instanciação de objetos. Posto isto, readapte a implementação da classe utilitária da questão anterior para que haja invocação da classe **`java.util.ArrayList<E>`** apenas no momento da instanciação das coleções, usando-se variáveis da interface **`java.util.Collection<E>`** para armazenamento das referências das instâncias e operações subsequentes de inserção. Por sua vez, para acesso sequencial e remoção dos elementos inseridos na primeira lista, deverá ser usado um iterador (a interface **`java.util.Collection<E>`** estabelece implementação de método, de nome **`iterator`**, que devolve um iterador sobre os itens da instância de coleção concreta que implementa aquela interface).

5. Considere a classe abaixo, para fins de armazenamento de dados de itens de uma típica agenda de contatos.

```
01  package lingprog2.lista04.questao05;
02
03  import java.text.SimpleDateFormat;
04  import java.util.GregorianCalendar;
05
06  public class Contato {
07
08      private String nome;
09      private long telefone;
10      private GregorianCalendar dataNascimento;
11
12          public Contato(String nome, long telefone, GregorianCalendar dataNascimento) {
13          this.nome = nome;
14          this.telefone = telefone;
```

```
15          this.dataNascimento = dataNascimento;
16      }
17
18      public String getNome() {
19          return nome;
20      }
21
22      public void setNome(String nome) {
23          this.nome = nome;
24      }
25
26      public long getTelefone() {
27          return telefone;
28      }
29
30      public void setTelefone(long telefone) {
31          this.telefone = telefone;
32      }
33
34      public GregorianCalendar getDataNascimento() {
35          return dataNascimento;
36      }
37
38      public void setDataAniversario(GregorianCalendar dataNascimento) {
39          this.dataNascimento = dataNascimento;
40      }
41
42      public String toString() {
43          return "Contato [nome=" + nome + ", "
44                  + "telefone=" + telefone + ", "
45                  + "dataNascimento=" + new
                      SimpleDateFormat("dd/MM/yyyy").format(dataNascimento.getTime()) + "]";
46      }
47
48  }
```

Implemente uma classe utilitária que disponha de método estático **`main`**, no qual seja manipulada uma lista de instâncias da classe **`Contato`** através de alguma classe da biblioteca de coleções da linguagem Java que implemente a interface **`java.util.List<E>`**. Deverá ser permitido a qualquer momento executar uma da seguintes operações: a) inserção de novo contato; b) listagem de contatos de determinado ano de nascimento já inseridos na lista; e c) encerramento do programa.